

Further heuristics for k -means: The merge-and-split heuristic and the (k, l) -means

Frank Nielsen
 Sony Computer Science Laboratories, Japan
 École Polytechnique, France
 Frank.Nielsen@acm.org

Richard Nock
 NICTA, Australia
 Richard.Nock@nicta.com.au

Abstract

The k -means clustering problem asks to partition the data into k clusters so as to minimize the sum of the squared Euclidean distances of the data points to their closest cluster center. Finding the optimal k -means clustering of a d -dimensional data set is NP-hard in general and many heuristics have been designed for minimizing monotonically the k -means objective function. Those heuristics got trapped into local minima and thus heavily depend on the initial seeding of the cluster centers. The celebrated k -means++ algorithm is such a randomized seeding method which guarantees probabilistically a good initialization with respect to the global minimum. In this paper, we first show how to extend Lloyd's batched relocation heuristic and Hartigan's single-point relocation heuristic to take into account empty-cluster and single-point cluster events, respectively. Those events tend to increasingly occur when k or d increases, or when performing several restarts of the k -means heuristic with a different seeding at each round in order to keep the best clustering in the lot. We show that those special events are a blessing because they allow to partially re-seed some cluster centers while further minimizing the k -means objective function. Second, we describe a novel heuristic, called merge-and-split k -means, that consists in merging two clusters and splitting this merged cluster again with two new centers provided it improves the k -means objective. Hartigan's heuristic can improve a Lloyd's heuristic when it reaches a local minimum, and similarly this novel heuristic can improve Hartigan's k -means when it has converged to a local minimum. We show empirically that this merge-and-split k -means improves over the Hartigan's heuristic which is the *de facto* method of choice. Finally, we propose the (k, l) -means objective that generalizes the k -means objective by associating the data points to their l closest cluster centers, and show how to either directly convert or iteratively relax the (k, l) -means into a k -means in order to reach better local minima.

1 Introduction

Clustering is the task that consists in grouping data into homogeneous clusters with the goal that intra-cluster data should be more similar than inter-cluster data. Let $\mathcal{P} = \{p_1, \dots, p_n\}$ be a set of n points¹ in \mathbb{R}^d . Let $\mathcal{C}_1, \dots, \mathcal{C}_k$ be the k *non-empty* clusters partitioning \mathcal{P} and denote by

¹For the sake of clarity and without loss of generality, we do not consider weighted points.

$\mathcal{K} = \{c_1, \dots, c_k\}$ the set of k cluster centers, the *cluster prototypes*. k -Means is one of the oldest and yet prevalent clustering technique that consists in minimizing:

$$e(\mathcal{P}, \mathcal{K}) = \sum_{i=1}^n \min_{j=1}^k D(p_i, c_j) = \sum_{i=1}^n D(p_i, c_{l_i}) = \sum_{j=1}^k \sum_{p \in \mathcal{K}_j} D(p, c_j), \quad (1)$$

where $D(p, q) = \|p - q\|^2$ denotes the *squared* Euclidean distance, and l_i the index (or label) of the center of \mathcal{K} that is the closest nearest neighbor to p_i (say, in case of ties, choose the minimum integer). Finding an optimal clustering minimizing globally $\min_{\mathcal{K}} e(\mathcal{P}, \mathcal{K})$ is NP-hard when $d > 1$ and $k > 1$ [21, 8], and polynomial when $d = 1$ using dynamic programming [4] or when $k = 1$ setting c to the center of mass. Note that there is an *exponential number* of optimal k -means clustering yielding the same optimal objective function: Indeed, consider an equilateral triangle with $n = 3$ and $k = 2$, we thus get 3 equivalent optimal clustering related by rotational symmetries. Then make s far away separated copies so that $n = 3s$ and consider $k = 2s$, we end up with $3^s = 3^{\frac{n}{2}}$ optimal k -means clustering. Minimizing the k -means function of Eq. 1 is equivalent to minimizing the sum of intra-cluster squared distances or maximizing the sum of inter-cluster squared distances:

$$\min_{\mathcal{K}} e(\mathcal{P}, \mathcal{K}) \equiv \min_{\mathcal{K}} \sum_{j=1}^k \sum_{p_i, p_j \in \mathcal{C}_j} \|p_i - p_j\|^2 \equiv \max_{\mathcal{K}} \sum_{j=1}^k \sum_{p_i \in \mathcal{C}_j, p_j \notin \mathcal{C}_j} \|p_i - p_j\|^2 \quad (2)$$

Many heuristics have been proposed to overcome the NP-hardness of k -means. They can be classified into two main groups: The *local search* heuristics and the *global* heuristics that can be used to initialize the local heuristics. For example, the following four heuristics are classically² implemented:

- **Forgy [10] (random):** Draw uniformly at random k points from \mathcal{P} to set the cluster prototypes \mathcal{K} inducing the partition. It can be proved that this best *discrete* k -means (with $\mathcal{K} \subset \mathcal{P}$) yields a 2-approximation factor compared to the ordinary k -means using a proof by contradiction based on the *variance-bias decomposition*: $e(\mathcal{P}, c') = v(\mathcal{P}) + n\|c' - c\|^2$, where $v(\mathcal{X}) = \sum_{i=1}^n \|p_i - c\|^2 = \sum_{i=1}^n \|p_i\|^2 - n\|c\|^2$ denotes the variance and $c = \frac{1}{n} \sum_{i=1}^n p_i$ the centroid. In fact, $e(\mathcal{P}, \mathcal{K}) = \sum_{j=1}^k v(\mathcal{C}_j)$, the sum of intra-cluster variances (and $e(\mathcal{P}, \mathcal{K}) = \sum_{i=1}^n \|p_i\|^2 - \sum_{j=1}^k n_j \|c_j\|^2$).
- **MacQueen [20] (online):** From a given initialization of the k centers defining singleton clusters (say, $\mathcal{C}_j = \{p_j\}$ for the k clusters), we add a new point at a time to the cluster that contains the closest center, update that cluster centroid, and reiterate until convergence. This heuristic is also called the online or single-point k -means [11].
- **Lloyd [19] (batched):** From a given initialization of cluster prototypes, (1) assign points to their closest cluster, (2) relocate cluster centers to their cluster centroids, and reiterate those two steps until convergence.
- **Hartigan [12, 13] (single-point relocation):** From a given initialization, find how to move a point from a cluster to another cluster so that the k -means cost of Eq. 1 strictly decreases and reiterate those single-point relocations until convergence is reached. Note that a point maybe assigned to a cluster which center is *not* its closest center [24].

²See for example the R language for statistical computing, <http://www.r-project.org/>

In general, a k -means clustering technique partitions the data into pairwise non-overlapping convex hulls $\text{CH}(\mathcal{C}_1), \dots, \text{CH}(\mathcal{C}_k)$: The *Voronoi partition*. A partition is said *stable* when a local improvement of the heuristic cannot improve its k -means score. Let $P_{F,Q,L,H}(n, k)$ denotes the maximum number of stable k -means partitions obtained by Forgy's, MacQueen's, Lloyd's and Hartigan's schemes, respectively.

Fact 1 (Voronoi partitions) *We have $P_F(n, k) \leq \binom{n}{k}$ and $P_H(n, k) \leq P_L(n, k) \leq P_{\text{CH}}(n, k) < < P(n, k)$, where $P(n, k) = \frac{1}{k!} \sum_{i=0}^k (-1)^{k-i} \binom{k}{i} i^n$ denotes the number of partitions of n elements into k non-empty subsets (that is, the Stirling numbers of the second kind) and $P_{\text{CH}}(n, k)$ denotes the number of partitions with non-overlapping (and non-empty) convex hulls (that is, the number of k -Voronoi partitions).*

Hartigan's single-point relocation heuristic may improved Lloyd's clustering but not the converse [23]. Note that Lloyd's heuristic may require an exponential number of iterations to converge [25]. It is an open question [24] to bound the maximum number of Hartigan's iterations.

On one hand, for those local heuristics performing pivots on Voronoi partitions using primitives, initialization (i.e., the initial Voronoi partition) is crucial [7] to obtain a good clustering, and several restarts, denoted by `mstart`, are performed in practice to choose the best clustering. In practice, Forgy's initialization has been replaced by k -means++ [2] which provides an *expected* $\bar{O}(\log k)$ competitive initialization. However, it was shown that there exists point sets (even in 2D) for which the probability to get such a good initialization is exponentially low [6] (and thus requiring exponentially many initialization restarts to reach a good Voronoi partition with high probability).

On the other hand, the *global k -means* [18, 26] builds incrementally the clustering by adding one seed at a time. Given a current s -clustering it chooses the point in \mathcal{P} that minimizes the $(s + 1)$ -means objective function. Thus initialization is limited to choosing the first point, and all points can be considered as this first starting point. However, Global k -means requires more computation.

In this paper, we do not address the problem of choosing the most appropriate number, k , of clusters: This model selection problem has been investigated in [22, 17]. We also consider the squared Euclidean distance although the results apply to any other Bregman divergence [3, 23].

The paper is organized as follows: We investigate the blessing of empty-cluster exceptions in Lloyd's heuristic in Section 2, and of single-point-cluster exceptions in Hartigan's scheme in Section 3. In Section 4, we describe our novel heuristic merge-split-cluster k -means and report on its performances with respect to Hartigan's heuristic. In Section 5, we present a generalization of the k -means objective function where each point is associated to its l closest clusters: the (k, l) -means clustering. We show how to directly convert or iteratively relax a sequence of (k, l) -means to a k -means and compare experimentally those solutions with a direct k -means. Finally, Section 6 wraps up the contributions and discusses further perspectives.

2 The blessing of empty-cluster exceptions in Lloyd's batched k -means

Lloyd's k -means [19] starts by initializing the seeds of the cluster centers $\mathcal{K} = \{c_1, \dots, c_k\}$, and then iterates by assigning the data to their closest cluster center with respect to the squared

Euclidean distance, and then relocates the cluster centers to their centroids. Those batched assignment/relocation iterations are repeated until convergence is reached: The k -means cost monotonically decreases with guaranteed convergence after a finite number of iterations [15]. The complexity of Lloyd's k -means is $O(ndks)$ where s denotes the number of iterations. It has been proved that Lloyd's k -means performs a maximum number s of iterations exponential [25] or polynomial in n , d and the spread³ of the point set [16]. Some 1D point set are reported to take $\Omega(n)$ iterations even for $k = 2$, see [11]. We first, report a lower bound on the number of Lloyd's stable optima $P_L(n, k)$:

Fact 2 (Exponentially many Lloyd's k -means minima) *Lloyd's k -means may have $P_L(n, k) = \Omega(2^{\frac{n}{2k}})$ stable local minima.*

The proof follows from the gadget illustrated in Figure 1.

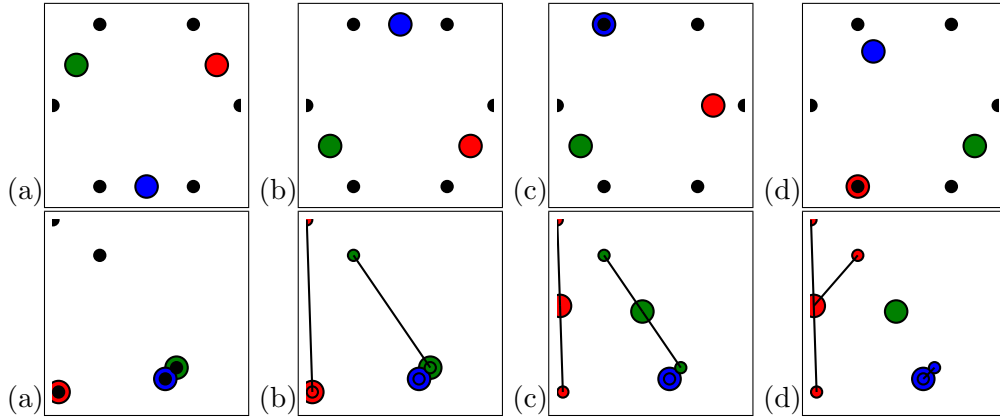


Figure 1: Top: Lloyd's k -means may have an exponential number of stable optima: Use locally the $k = 2p$ -gon (here $p = 3$) gadget that admits 2 global solution (a) and (b). Lloyd's k -means can be trapped into a local minimum: Cost in (c) and (d) is ~ 0.5417 compared to the global minima 0.375) in (a) and (b). Centroids are depicted by large colored disks. Bottom: Lloyd's k -means local optimization technique may produce *empty cluster exceptions*. Consider $n = 5$ points and $k = 3$ clusters: $p_1 = (0, 0)$, $p_2 = (0.25, 0.19)$, $p_3 = (0.03, 0.92)$, $p_4 = (0.66, 0.79)$ and $p_5 = (0.6, 0.85)$ with $k = 3$ and "random" Forgry initialization: $c_1 = p_3, c_2 = p_4$ and $c_3 = p_5$. Then the initial k -means cost (a) is 1.3754, the first iteration (b) and (c) yields cost 0.6877 and then at the second iteration we have an empty cluster exception in (d): The green cluster.

³The spread Δ is the ratio of the maximum point inter-distance over the minimum point inter-distance.

Data: $\mathcal{P} = \{(w_1, p_1), \dots, (w_n, p_n)\}$ a data set of size n , $k \in \mathbb{N}$: number of clusters

Result: A clustering partition $\mathcal{C}_1, \dots, \mathcal{C}_k$ where each point belongs to exactly one cluster (hard membership)

Initialization: Get k cluster centers $\mathcal{C} = \{c_1, \dots, c_k\}$ by choosing cluster prototypes at random from \mathcal{P} (e.g., Forgy or k -means++);

Iter $\leftarrow 0$;

while *not converged* **do**

Increment Iter, $e = 0$;

(a) Assign each point p_i to its closest cluster \mathcal{C}_{a_i} ;

/* iNN denotes the index of the nearest neighbor */

$a_i = \text{iNN}(p_i; \mathcal{K}).$

(b) Relocate each cluster prototype c_j by taking the center of mass of its assigned points;

$\mathcal{C}_j = \{p \in \mathcal{P} : j = \text{iNN}(p; \mathcal{C})\}, \quad n_j = \sum_{p_l \in \mathcal{C}_j} w_l.$

if $n_j > 0$ **then**

Non-empty cluster and centroid relocation:

$$c_j = \frac{1}{n_j} \sum_{p_l \in \mathcal{C}_j} w_l p_l$$

else

$e \leftarrow e + 1$;

end

(c) New seeding ;

/* Empty cluster exception (may have occurred overall $\Omega(k)$ times) */

1 Choose e new seeds for the empty clusters using k -means++ or global k -means, etc.;

Check for convergence by checking if at least one a_i is different from the previous iteration;

if Iter $>$ maxIter **then**

break;

end

end

Algorithm 1: Extended Lloyd’s k -means clustering: batched updates handling empty cluster exceptions.

The Hartigan’s heuristic [12, 13] proceeds by relocating a single point between two clusters provided that the k -means cost function decreases. It can thus further decrease the k -means score when Lloyd’s batched algorithm is stuck into a local minimum (but not the converse). Recently, Hartigan’s heuristic [23] was suggested to replace Lloyd’s heuristic on the basis that Hartigan’s local minima is a subset of Lloyd’s optima (Theorem 2.2 of [24]). We argue that this is true only when no *Empty Cluster Exceptions* (ECEs) are met by Lloyd’s iterations. Figure 1 illustrates a toy data set where Lloyd’s k -means meets such an empty-cluster exception. In general at the end of the relocation stage, when points are assigned to their closest current centroids, we may have some

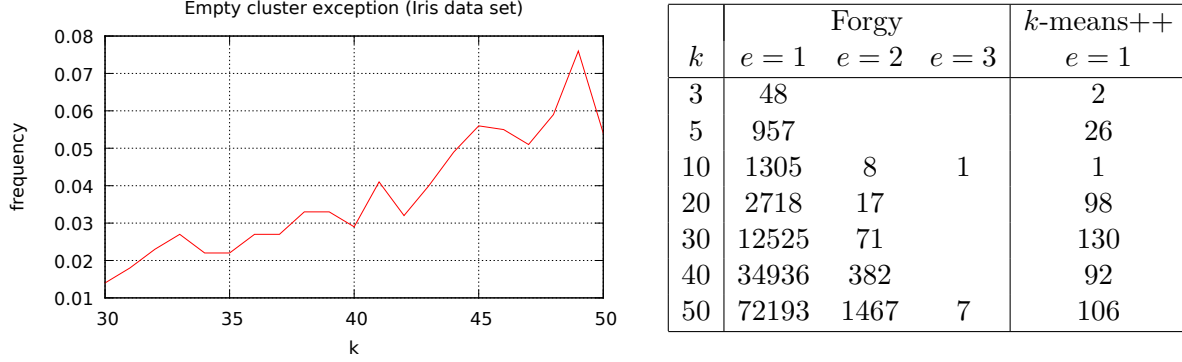


Figure 2: Left: Graph plot of the frequency of empty-cluster exceptions ($e > 0$) for Lloyd’s k -means using Forgý’s initialization on the normalized *Iris* data set computed by averaging over a million runs. Right: Number of ECEs depend on the initialization method: At $k = 50$, we observe a frequency of 7.2% for one empty cluster, 0.014% for two empty clusters, etc. for Forgý’s seeding but k -means++ initialization produces less such exceptions.

empty clusters.

Fact 3 (empty-cluster exceptions) *Lloyd’s batched k -means may produce $e = \Omega(k)$ empty cluster exceptions in a round.*

Proof follows from Figure 1 by creating $s = n/5$ far apart (non-interacting) copies of the gadget and setting $k = 3s$.

However, those empty-cluster exceptions are a *blessing* because we may add e new seeds that will further decrease significantly the cost of k -means: This is a partial re-seeding. Thus the extended Lloyd’s heuristic is: (a) assignment, (b) relocation, and (c) partial re-seeding to keep exactly k non-empty clusters for the next stage. We may use various heuristics for partially re-seeding like the incremental global k' -means [26] starting from $k' = k - e$ to $k' = k$, etc.

To evaluate the frequency at which those empty-cluster exceptions occur and their number e , let us take the *Iris* data set from the UCI repository [1]: It consists of $n = 351$ samples with $d = 4$ features (classified into $k = 3$ labels) that we first renormalize the data-set so that coordinates on each dimension have zero mean and unit standard deviation. Let us run Lloyd’s k -means with (Forgý’s) random seed initialization (with a maximum number of 1000 iterations) for $mstart = 1000000$. We count the number of empty cluster exceptions and report their frequency in the graph of Figure 2. We observe that the larger the k , and the more frequent the exceptions. This phenomenon was also noticed in [5]. Furthermore, e increases with the dimension d too [5]. However, note that this is a tendency and the number of empty-cluster exceptions vary a lot from a data set to another one (given an initialization heuristic).

Let us now run $mstart = 1000000$ k -means and report the *empirical frequency* of having $e = 1, 2, 3, \dots$ *simultaneous* empty-cluster exceptions. (Note that our replicated toy data-sets of Figure 1 may provide $\Omega(k)$ values). The empty-cluster frequency depends on the initialization scheme: It is higher when using Forgý’s heuristic and lower when using k -means++ or global k -means. Table 2 demonstrates empirically this observation. As noticed in [5], the number of cluster-empty exceptions rise with k and d and the authors [5] avoided this problem by setting minimum input

k /method	classic Lloyd		Lloyd+partial reseeding		#ECEs
	avg	min	avg	min	
30	12.86	9.88	12.86	9.88	7685
40	9.72	7.30	9.72	7.28	23633
50	7.5	5.47	7.5	5.47	55726

Table 1: Comparing Lloyd’s k -means heuristics without or without partial reseeding (Forgy) when meeting empty-cluster exceptions on **Iris** dataset with a million restart using the same Forgy’s initialization at each round. Observe that some better local minima are reached when using partial reseeding at empty-cluster exceptions.

size on clusters. They surprisingly show empirically that k -means with constraints gave better clustering than k -means without constraints in practice!

Finally, let us compare the best minimum k -means score when performing Lloyd’s heuristic (and stopping when we meet an empty cluster exception), and the extended Lloyd’s heuristic that partially reseeds the current clustering when the algorithm meets empty-cluster exceptions. Partial reseeding can be done in many ways by starting from the current number of cluster centers the usual seeding methods (Forgy, k -means++ or global k -means). Table 1 presents the results for the proof of concept using Forgy’s re-seeding: We observe that partial reseeding at ECEs allows to reach (slightly) better local minima (see $k = 40$ in Table 1).

3 The blessing of single-point cluster exceptions in Hartigan’s heuristic

Hartigan’s heuristic [24] consider relocating a single-point provided that it decreases the k -means objective function. In [23], a synthetic noisy data-set is built so that with probability tending to 1 (as the dimension tends to infinity) any initial random partition is stable wrt. Lloyd’s k -means while Hartigan’s converges to the correct solution. We recall that Hartigan’s local minima are a subset of Lloyd’s minima [24] *provided* that Lloyd’s heuristic did not encounter empty-cluster exceptions. Note that a single-point cluster (with associated cluster having zero variance) cannot be relocated to other clusters since it necessarily increases the k -means energy (sum of intra-cluster variances):

$$e(\mathcal{P}, \mathcal{K}) = \sum_{j=1}^k v(\mathcal{C}_j) = \sum_{i=1}^n \|p_i\|^2 - n_j \sum_{j=1}^k \|c_j\|^2, \quad \sum_{j=1}^k n_j = n. \quad (3)$$

Table 2 provides statistics on the Hartigan’s k -means score and the number of single-point-cluster exceptions (SPCEs) met when performing Hartigan’s heuristic.

Consider the case of *Single-point-Cluster Exceptions (SCEs)* in Hartigan’s scheme where we decide to merge this single-point cluster $\mathcal{C}_i = \{x\}$ with another cluster \mathcal{C}_j and redraw another center from \mathcal{P} (that can thus decrease significantly the variance of the change cluster). We accept this relocation iff. this merge&re-seed operation decreases the k -means loss. For example, when $k = 30$ (and $mstart = 1000$), the classical Hartigan’s best clustering has k -means score 9.75 while the heuristic with partial reseeding (associating the single-point clusters to their closest other clusters),

k	Hartigan's k -means			Single-point cluster exceptions		
	min	avg	max	min	avg	max
30	9.74	11.28	15.66	3	20893.27	34007
35	8.20	9.48	13.27	6	43700.20	75911
40	6.98	8.06	12.69	12	61437.81	103407
45	5.79	6.92	11.23	9	83113.54	163344
50	5.06	5.95	8.96	13	204222.78	367437

Table 2: Some statistics on Hartigan's heuristic on the `Iris` data set: min/avg/max k -means score and min/avg/max number of single-point cluster exceptions (SPCEs).

we obtain 9.65. We keep the experiments short here since the next Section improves Hartigan's heuristic with detailed experiments.

4 A novel heuristic: The merge-and-split-cluster k -means

This novel heuristic proceeds by considering pairs of clusters $(\mathcal{C}_i, \mathcal{C}_j)$ with corresponding centers c_i and c_j . The *basic local search primitive* (pivot) consists in computing the best k -means score difference by *merging and splitting again* $\mathcal{C}_{i,j} = \mathcal{C}_i \cup \mathcal{C}_j$ with two new centers c'_i and c'_j . Let \mathcal{C}'_i and \mathcal{C}'_j denote the *Voronoi partition* of $\mathcal{C}_{i,j}$ induced by c'_i and c'_j . Since the clusters other than \mathcal{C}_i and \mathcal{C}_j are untouched, the difference of the k -means score is written as:

$$\Delta(\mathcal{C}_i, \mathcal{C}_j) = e_1(\mathcal{C}_i, c_i) + e_1(\mathcal{C}_j, c_j) - (e_1(\mathcal{C}'_i, c'_i) + e_1(\mathcal{C}'_j, c'_j)), \quad (4)$$

where $e_1(\mathcal{C}, c)$ denotes the 1-means objective function: namely, the cluster variance of \mathcal{C} with respect to center c . There are several ways (randomized or deterministic) to implement the merge-and-split operation: For example, the two new centers can be found by computing:

- a 2-means: A brute-force method computes all hyperplanes⁴ passing through $d+1$ (extreme) points and the induced sum of variances of the below-above clusters in $O(n^{d+2})$ -time. Using topological sweep [15], it can be reduced to $O(n^d)$ time. Note that for $k=2$ and unfixed d , 2-means is NP-hard [8]. We can also use coresets to get a $(1+\epsilon)$ -approximation of a 2-means [9] in linear time $O(nd)$.
- a discrete 2-means: We choose among the $n_{i,j} = n_i + n_j$ points of $\mathcal{C}_{i,j}$ the two best centers (naively implemented in $O(n^3)$). This yields a 2-approximation of 2-means.
- a 2-means++ heuristic: We pick c'_i at random, then pick c'_j randomly according to the normalized distribution of the squared distances of the points in $\mathcal{C}_{i,j}$ to c'_i , see k -means++ [2]. We repeat a given number α of rounds this initialization (say, $\alpha = 1 + 0.01 \binom{n_{i,j}}{2}$) and keeps the best one.

When $\Delta(\mathcal{C}_i, \mathcal{C}_j) > 0$, we accept replacing (\mathcal{C}_i, c_i) and (\mathcal{C}_j, c_j) by (\mathcal{C}'_i, c'_i) and (\mathcal{C}'_j, c'_j) , respectively. Otherwise, we consider another pair of clusters and stop iterating when all pairs do not produce

⁴We do not need to compute explicitly the equation of the hyperplane since clockwise/counterclockwise *orientation predicates* are used instead. Those predicates rely on computing the sign of a $(d+1) \times (d+1)$ matrix determinant.

Data set	Hartigan		Discrete Hartigan		Merge&Split	
	cost	#ops	cost	#ops	cost	#ops
Iris(d=4,n=150,k=3)	112.35	35.11	101.69	33.54	83.95	31.36
Wine(d=13,n=178,k=3)	607303	97.88	593319	100.02	570283	100.47
Yeast(d=,n=1484,k=10)	47.10	1364.0	57.34	807.83	50.20	190.58

Data set	Hartigan++		Discrete Hartigan++		Merge&Split++	
	cost	#ops	cost	#ops	cost	#ops
Iris(d=4,n=150,k=3)	101.49	19.40	90.48	18.93	88.56	8.84
Wine(d=13,n=178,k=3)	3152616	18.76	2525803	24.61	2498107	9.67
Yeast(d=8,n=1484,k=10)	47.41	1192.38	54.96	640.89	51.82	66.30

Table 3: Average performance over 1000 trials of the merge-and-split k -means heuristic compared to Hartigan’s and discrete Hartigan’s heuristics. Top: Common Forgy’s initialization and the MSC k -means has been implemented using an optimal discrete 2-means. Bottom: Common k -means++ initialization and the MSC k -means has been implemented using a 2-means++ with $\alpha = 0.01\%$. We observe experimentally that MSC heuristic yields always better performance than Hartigan’s discrete single-point relocation heuristic, and is often significantly better than Hartigan’s heuristic. Note that k -means++ seeding performs better than Forgy’s seeding

a lower k -means score. This heuristic can be classified as a macro kind of Hartigan-type heuristic that is *not* based on local Voronoi assignment. Indeed, Hartigan’s heuristic moves a point x from a cluster \mathcal{C}_i to a cluster \mathcal{C}_j and update the two centroids correspondingly. Our heuristic also change these two clusters but can accept further improvements with respect to a 2-means operation on $\mathcal{C}_{i,j}$. Thus at the last stage of a Hartigan’s heuristic, we can perform this merge-and-split heuristic to further improve the clustering. (This heuristic can further be generalized by simultaneous merging-and-splitting of r clusters.)

Theorem 1 *The merge-and-split k -means heuristic decreases monotonically the objective function and converges after a finite number of iterations.*

Since each pivot step between Voronoi partitions strictly decreases the k -means score $e(\mathcal{P}, \mathcal{K}) \geq 0$ by $\Delta(\mathcal{C}_i, \mathcal{C}_j) > 0$ and that $\min_{\mathcal{C}_i, \mathcal{C}_j} \Delta(\mathcal{C}_i, \mathcal{C}_j) > 0$ is lower bounded, it follows that the merge-and-split k -means converges after a finite number of iterations. We compare our heuristic with both Hartigan’s ordinary and discrete variants that consists in moving a point to another cluster iff. the two recomputed medoids of the selected clusters yield a better k -means score. Heuristic performances are compared with the same initialization (Forgy’s or k -means++ seeding) and by averaging over a number of rounds: Observe in Table 3 that our heuristic (MSC for short) always outperforms discrete Hartigan’s method not suprisingly. Although the number of basic primitives (#ops) is lower for MSC, each such operation is more costly. Thus MSC k -means is overall more time consuming but gets better local optima solutions. Note that the discrete 2-means medoid splitting procedure is very well suited for the k -modes algorithm [14], a k -means extension working on categorical data sets.

5 Clustering with the (k, l) -means objective function

Let us generalize the k -means objective function as follows: For each data $p_i \in \mathcal{P}$, we associate p_i to its l nearest cluster centers $\text{NN}_l(p_i; \mathcal{K})$ (with iNN_l denoting the cluster center indexes), and ask to minimize the following (k, l) -means objective function (with $1 \leq l \leq k$):

$$e(\mathcal{P}, \mathcal{K}; l) = \sum_{i=1}^n \sum_{a \in \text{iNN}_l(p_i; \mathcal{K})} \|p_i - c_a\|^2. \quad (5)$$

When $l = 1$, this is exactly the k -means objective function of Eq. 1. Otherwise the clusters overlap and $|\cup_{j=1}^k \mathcal{C}_j| = nl$. Note that when $l = k$, since $\text{NN}_k(p_i; \mathcal{K}) = \mathcal{K}$ all cluster centers c_1, \dots, c_k coincide to the *centroid* $\bar{p} = \frac{1}{n} \sum_i p_i$ (or barycenter), the *center of mass*. We observe that:

Fact 4 $e(\mathcal{P}, \mathcal{K}; l) \geq l \times e(\mathcal{P}, \mathcal{K}; 1)$ with equality reached when $l = k$.

Both Lloyd’s and Hartigan’s heuristics can be adapted straightforwardly to this setting.

Theorem 2 *Lloyd’s (k, l) -means decreases monotonically the objective function and converge after a finite number of steps.*

Proof: Let c_{2t} and c_{2t+1} denote the cost at round t , for the assignment (c_{2t}) and relocation (c_{2t+1}) stages. Let c_0 be the initial cost (say, from Forgy’s initialization of \mathcal{K}^0). For $t > 0$, we have: In the assignment stage $2t$, each *point* p_i is assigned to its l nearest neighbor centers $\text{NN}_l(p_i; \mathcal{K}^{t-1})$. Therefore, we have $c_{2t} = \sum_{i=1}^n \sum_{c \in \text{NN}_l(p_i; \mathcal{K}^{t-1})} D(p_i, c) \leq c_{2t-1}$. In the relocation stage $2t + 1$, each cluster \mathcal{C}_j^t is updated by taking its centroid c_j^{t+1} . Thus we have $c_{2t+1} = \sum_{j=1}^k \sum_{p \in \mathcal{C}_j^t} D(p, c_j^{t+1}) \leq \sum_{j=1}^k \sum_{p \in \mathcal{C}_j^t} D(p, c_j^t) \leq c_{2t}$. When $\mathcal{K}^{t+1} = \mathcal{K}^t$ (and thus $c_{2t} = c_{2t-1}$), we stop the batched iterations.

Figure 3 illustrates a $(k, 2)$ -means on a toy data-set.

Since $c_t \geq 0$ for all t and the iterations strictly decreases the score function, the algorithm converges. Moreover, since the number of different cluster sets induced by (k, l) means is upper bounded by $O(n^{kl})$, and that cluster sets cannot be repeated, it follows that (k, l) -means converges after a *finite number* of iterations. The bound can further be improved by considering the l -order weighted Voronoi diagrams, similarly to [15]. Note that the basic Lloyd’s (k, l) -means may also produce empty-cluster exceptions although those become rarer as l increase (checked experimentally).

Although (k, l) -means is interesting in its own (see Discussion in Section 6), it can also be used for k -means. Indeed, instead of running a local search k -means heuristic that may be trapped too soon into a “bad” local minimum, we prefer to run a (k, l) means for a prescribed l . We can then convert a (k, l) -means by assigning to each point p_i its closest neighbor (among the l assigned at the end of the (k, l) -means), and then compute the centroids and launch a regular Lloyd’s k -means to finalize: Let $(k, l) \downarrow$ -means denote this conversion. For example, for $k = 6$ and $l = 2$, the converted $(k, 2)$ -means beats the k -means 80% of the time for $\text{mstart} = 10000$ using Forgy’s initialization on *Iris*. Table 4 shows experimentally that converted $(k, 2)$ -means beats on *average* the regular k -means (for the *Iris* data-set) and this phenomenon increases not surprisingly with k . However the best minimum score is often obtained by classical k -means. Thus it suggests that (k, l) performs better when the number of restarts is limited. In fact, (k, l) -means tend to smooth the k -means optimization landscape and produce less local minima but also smooth the best minima.

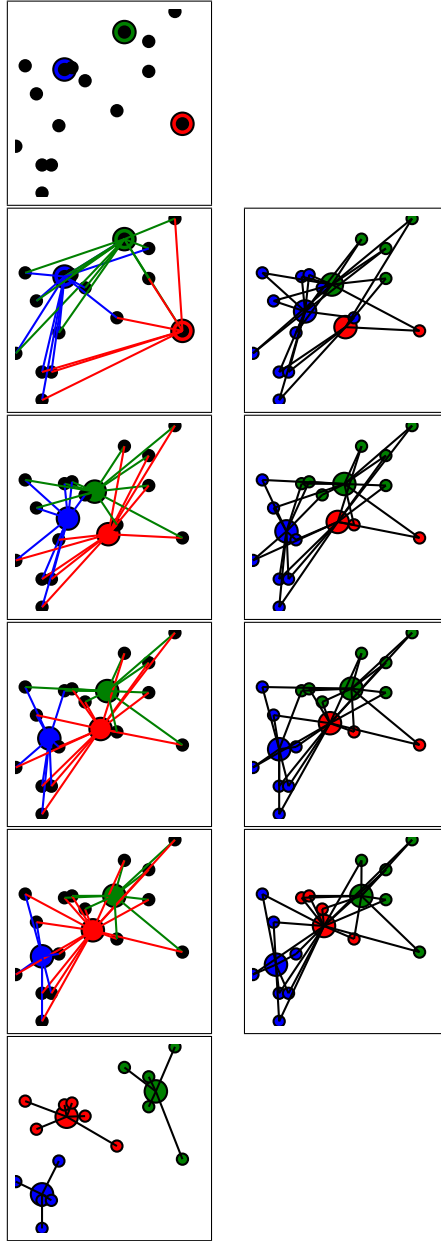


Figure 3: $(k, 2)$ -means: Each data point is associated to its *two* closest cluster center neighbors. After converging, we relax the $(k, 2)$ -means solution by keeping only the closest neighbor on the current centroids and run the classic k -means. Alternatively, we can relax iteratively the (k, m) means into a $(k, m - 1)$ -means until we get a k -means.

k	win	k -means		$(k, 2)$ \downarrow -means	
		min	avg	min	avg
3	20.8	78.94	92.39	78.94	78.94
4	24.29	57.31	63.15	57.31	70.33
5	57.76	46.53	52.88	49.74	51.10
6	80.55	38.93	45.60	38.93	41.63
7	76.67	34.18	40.00	34.29	36.85
8	80.36	29.87	36.05	29.87	32.52
9	78.85	27.76	32.91	27.91	30.15
10	79.88	25.81	30.24	25.97	28.02

k	l	win	k -means		$\downarrow (k, l)$ -means	
			min	avg	min	avg
5	2	58.3	46.53	52.72	49.74	51.24
5	4	62.4	46.53	52.55	49.74	49.74
8	2	80.8	29.87	36.40	29.87	32.54
8	3	61.1	29.87	36.19	32.76	34.04
8	6	55.5	29.88	36.189	32.75	35.26
10	2	78.8	25.81	30.61	25.97	28.23
10	3	82.5	25.95	30.23	26.47	27.76
10	5	64.7	25.90	30.32	26.99	28.61

Table 4: Comparing k -means with $(k, 2)$ \downarrow -means (left) and with $\downarrow (k, l)$ -means (right). The percentage of times it outperforms k -means is denoted by win.

We can also perform a cascading conversion of (k, l) -means to k -means: Once a local minimum is reached for (k, l) -means, we initialize a $(k, l - 1)$ means by dropping for each point p_i its farthest cluster, perform a Lloyd’s $(k, l - 1)$ -means, and we reiterate this scheme until we get a $(k, 1)$ -means: An ordinary k -means. Let $\downarrow (k, l)$ -means denote this scheme. Table 4 (right) presents the performance comparisons of a regular Lloyd’s k -means with a Lloyd’s $(k, l \downarrow 1)$ -means for various values of l with the initialization of both algorithms performed by the same seeding for fair comparisons.

6 Discussion

We have extended the classical Lloyd’s and Hartigan’s heuristics with partial re-seeding and proposed new local heuristics for k -means. We summarize our contributions as follows: First, we showed the *blessing* of empty-cluster events in Lloyd’s heuristic and of single-point-cluster events in Hartigan’s heuristic. These events happen increasingly when the number of cluster k or the dimension d increase, or when running those heuristics a given number of trials to choose the best solution. Second, we proposed a novel *merge-and-split-cluster k -means heuristic* that improves over Hartigan’s heuristic that is currently the *de facto* method of choice [23]. We showed experimentally that this method brings better k -means result at the expense of computational cost. Third, we generalized the k -means objective function to the (k, l) -means objective function and show how to directly convert or iteratively relax a (k, l) -means heuristic to a k -means avoiding potentially being trapped into too many local optima. (k, l) -Means is yet another *exploratory* clustering technique for browsing the space of hard clustering partitions. For example, when k -means is trapped, we may consider a (k, l) -means to get out of the local minimum and then convert the (k, l) -means to a k -means to explore a new (local) minimum.

References

- [1] D.J. Newman A. Asuncion. UCI machine learning repository, 2007.
- [2] D. Arthur and S. Vassilvitskii. k -means++ : the advantages of careful seeding. In *SODA*, pages 1027 – 1035, 2007.

- [3] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.
- [4] R. Bellman. A note on cluster analysis and dynamic programming. *Mathematical Biosciences*, 18(3-4):311 – 312, 1973.
- [5] K. Bennett, Paul S. Bradley, and Ayhan Demiriz. Constrained k -means clustering. MSR-TR-2000-65, 2000.
- [6] A. Bhattacharya, R. Jaiswal, and N. Ailon. A tight lower bound instance for k -means++ in constant dimension. In *Theory and Applications of Models of Computation*, LNCS 8402, pages 7–22, 2014.
- [7] S. Bubeck, M. Meila, and U. von Luxburg. How the initialization affects the stability of the k -means algorithm. *ESAIM: Probability and Statistics*, 16:436–452, 1 2012.
- [8] S. Dasgupta. The hardness of k -means clustering. CS2007-0890, University of California, USA, 2007.
- [9] D. Feldman, M. Monemizadeh, and C. Sohler. A PTAS for k -means clustering based on weak coresets. In *SoCG*, pages 11–18. 2007.
- [10] E. W. Forgy. Cluster analysis of multivariate data: efficiency vs interpretability of classifications. *Biometrics*, 1965.
- [11] S. Har-Peled and B. Sadri. How fast is the k -means method? In *SODA*, pages 877–885. SIAM, 2005.
- [12] J. A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 99th edition, 1975.
- [13] J. A. Hartigan and M. A. Wong. Algorithm AS 136: A k -means clustering algorithm. *Journal of the Royal Statistical Society. Series C*, 28(1):100–108, 1979.
- [14] Z. Huang. Extensions to the k -means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.*, 2(3):283–304, September 1998.
- [15] M. Inaba, N. Katoh, and H. Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k -clustering. In *SoCG*, pages 332–339, 1994.
- [16] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. Piatko, R. Silverman, and A. Y Wu. The analysis of a simple k -means clustering algorithm. In *SoCG*, pages 100–109. 2000.
- [17] B. Kulis and M. I. Jordan. Revisiting k -means: New algorithms via Bayesian nonparametrics. In *ICML*, 2012.
- [18] A. Likas, N. Vlassis, and J. J Verbeek. The global k -means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.
- [19] S. P. Lloyd. Least squares quantization in PCM. Technical report, Bell Laboratories, 1957. reprinted in *IEEE Transactions on Information Theory*, March 1982.

- [20] J. B. MacQueen. Some methods of classification and analysis of multivariate observations. *Proceedings Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
- [21] M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k -means problem is NP-hard. In *WALCOM: Algorithms and Computation*, pages 274–285. Springer, 2009.
- [22] D. Pelleg and A. W. Moore. X -means: Extending k -means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, 2000.
- [23] N. Slonim, E. Aharoni, and . Crammer. Hartigan’s k -means versus Lloyd’s k -means: Is it time for a change? In *IJCAI*, pages 1677–1684, 2013.
- [24] M. Telgarsky and A. Vattani. Hartigan’s method: k -means clustering without Voronoi. In *International Conference on Artificial Intelligence and Statistics*, pages 820–827, 2010.
- [25] A. Vattani. k -means requires exponentially many iterations even in the plane. *Discrete & Computational Geometry*, 45(4):596–616, 2011.
- [26] J. Xie, S. Jiang, W. Xie, and X. Gao. An efficient global k -means clustering algorithm. *Journal of computers*, 6(2), 2011.